ANALYSIS OF THE COUPLED FREQUENCIES AND MODE SHAPES OF VEHICLE CONTROL SYSTEMS

by

Yudell L. Luke
Rosemary Moran

FINAL REPORT
15 April 1965 - 14 April 1966

Contract NAS8-20139

MRI Project No. 2866-P

For

Aero-Astrodynamics Laboratory
National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Huntsville, Alabama 35812
Attn: Purchasing Office
PR-EC

MRI
MIDWEST RESEARCH INSTITUTE
425 VOLKER BOULEVARD/KANSAS CITY, MISSOURI 64110/AC 816 LO 1-0202

PREFACE

This report covers research initiated by the Aero-Astrodynamics Laboratory, NASA, George C. Marshall Space Flight Center, Huntsville, Alabama, under Contract NAS8-20139. Mr. Mario Rheinfurth of this laboratory served as project monitor.

The authors acknowledge with thanks the assistance of Messrs. Wyman Fair and Jet Wimp.

Approved for:

MIDWEST RESEARCH INSTITUTE

Sheldon L. Levy, Director
Mathematics and Physics Division

19 April 1966

lw

TABLE OF CONTENTS

SUMMARY

26652

In the design of missile control systems, an analysis of the coupled frequencies and mode shapes leads to the study of the generalized eigenvalue problem $D(s)x = 0$ where $D(s)$ is an $n \times n$ matrix whose elements are polynomials in s and x is a vector. In this report, we develop a systematic procedure for the evaluation of s and the corresponding vector x. A FORTRAN II-D program for the IBM 1620 computer is appended to produce the complete solution. A numerical example is provided.

I. INTRODUCTION

In the theory of linear vibrations of damped systems, one is naturally led to a matrix differential equation of the form

$$\left(A \frac{d^2}{dt^2} + B \frac{d}{dt} + C\right) x(t) = b(t) \tag{1.1}$$

where A, B and C are nxn matrices with elements independent of t , and x(t) and b(t) are 1xn column vectors. To compose the solution of (1.1), the solutions of the homogeneous system are required. These solutions are proportional to e^{st} whence we get the nonlinear eigenvalue equation

$$(As^2+Bs+C)\ x = 0 \tag{1.2}$$

where s is the eigenvalue and x its corresponding eigenvector. In this report we develop algorithms to find s and the corresponding x . Equation (1.2) can be generalized to the form

$$D(s)x = 0 \tag{1.3}$$

where D(s) is a nxn matrix whose elements are arbitrary polynomials in s. In the analysis of coupled frequencies and mode shapes of space vehicle control systems, a system like (1.3) may arise. Though our ideas for the solution of (1.2) can carry over for the solution of (1.3), our principal concern in the sequel is techniques for the solution of (1.2).

II. SELECTION OF TECHNIQUES

There are numerous possibilities for the solution of (1.2). For example, if one could hypothesize that the damping in each mode is light, say less than 10 per cent of critical, then it would appear feasible to design an iterative procedure based on a perturbation of the linear system $(As^2+C)x = 0$. One difficulty is that the damping may not be small. Indeed in the plan of space vehicles, the control equation can be so designed that, in effect, large amounts of damping can be present in various modes. There is yet another difficulty. In a known iterative procedure [1], if the suggested algorithm for the evaluation of s converges, it is known that the convergence is at least quadratic. Further, a theorem is available to establish a convergence neighborhood, that is, a neighborhood of the true s within which the iteration procedure converges. However, the radius of this neighborhood is not readily reduced to an a priori form from the usual input data. Iteration procedures have other disadvantages. For example, only one eigenvalue can be found at a time, and each step of the iterative procedure requires the inversion of an $n\times n$ matrix.

In view of the convergence uncertainty and other points mentioned above, it appears desirable to develop a direct (that is, noniterative) solution for the nonlinear eigenvalue problem. To facilitate our understanding of the equations which follow, we give a short summary of the ideas involved. Suppose that in (1.2), the coefficient of B is replaced by s_o. Under the assumption that A is nonsingular, we can write

$$(Is^2+F)x = 0 \ , \ F = A^{-1}\,(Bs_o+C) \quad , \tag{2.1}$$

where I is the identity matrix. We next seek the characteristic equations corresponding to F. Note that this is not the characteristic equation for our original problem (1.2) unless, of course, s_o is an eigenvalue of (1.2). We return to this point later. For now we remark that several procedures are available to evaluate directly the characteristic equation for (2.1). Most of these can be sensitive to the special peculiarities of the matrices and vectors involved. For example, degeneracies can occur when certain quantities required for division are null or nearly so. We propose to get the characteristic equation for (2.1) by a method due to Leverrier as modified by Faddeev [2,3]. See Fettis [4] for an exposition of the method together with an example. Though this procedure requires more operations than those known by the names of Krylov, Danilevsky, Samuelson and Bryan, see the references above and also Householder [5], it is utterly insensitive to the peculiarities just noted since no divisions are required, only multiplication and addition of matrices. From this point of view, the method of Leverrier is universal.

The characteristic equation for (2.1) can be written in the form

$$\sum_{k=0}^{n} (-)^k b_{n-k} \lambda^k \ , \ b_o = 1 \ , \ \lambda = -s^2 \ . \tag{2.2}$$

This last equation can also be used to represent the characteristic equation for (1.2). In this event, it is easy to show that b_r is a polynomial in s of degree r. That is, we can write

$$b_r = \sum_{m=0}^{r} g_{r-m}^{(r)} s^{r-m} \tag{2.3}$$

Now for each selected value of s_o, we compute a value b_r. If we compute b_r for $(r+1)$ distinct values of s_o, then by use of the Lagrangian interpolation formula, we can evaluate the coefficients $g_m^{(r)}$. As r runs through the sequence $1,2,\ldots n$, we require $(n+1)$ values of s_o. Thus, the method of Leverrier must be repeated $(n+1)$ times corresponding to $(n+1)$ distinct values of s_o.

It calls for remark that if the determinant of (As^2+Bs+C) is evaluated for $(2n+1)$ distinct values of s, then the Lagrangian interpolation method could be used to compute the coefficients of the characteristic equation. In this event, one could dispense with the developments surrounding (2.1). We have not experimented with this approach, for in some past considerations we have found that round-off errors can seriously affect the accuracy of the polynomial produced by the Lagrangian method as applied to high degree systems. The suggested procedure tends to minimize this difficulty.

Once the characteristic equation is established, its roots are easily determined by an iterative method due to Bairstow [6]. The corresponding vectors are then determined by solution of the linear equation system (1.2) or we may use (2.1) with s_o replaced by the characteristic value s.

We now turn to an exposition of the equations used to accomplish the various steps required in the solution of (1.2).

III. THE METHOD OF LEVERRIER

We have the linear system

$$Fx = \lambda x \tag{3.1}$$

where in the notation of the previous section

$$F = A^{-1}\,(Bs_o+C)\;,\;\lambda = -s^2\;. \tag{3.2}$$

The characteristic equation for (3.1) is expressed as

$$\varphi(\lambda) = \sum_{k=0}^{n} (-)^k b_{n-k}\lambda^k\;,\; b_o = 1 \quad . \tag{3.3}$$

Thus given the matrix F, we seek the coefficients b_k. We first present a summary of the procedure. Let d_{ij}, $i,j = 1,2,\ldots,n$ be the elements of a matrix D. That is,

$$D = (d_{ij}) \tag{3.4}$$

Then by the trace of D, written $(\mathrm{Tr}D)$ we mean

$$(\mathrm{Tr}D) = \sum_{k=0}^{n} d_{ii} \tag{3.5}$$

Sometimes the word "spur" is used instead of trace. We define matrices F_k, G_k and scalars q_k as follows.

$$F_1 = F\;,\; q_1 = (\mathrm{Tr}F_1)\;,\; G_1 = q_1 I - F_1$$

$$F_2 = FG_1,\; q_2 = (\mathrm{Tr}F_2)/2\;,\; G_2 = q_2 I - F_2$$

— — — — — — — — — — — — —

$$F_{n-1} = FG_{n-2}, q_{n-1} = (\mathrm{Tr}F_{n-1})/(n-1)\;,\; G_{n-1} = q_{n-1}I - F_{n-1}$$

$$F_n = FG_{n-1}\;,\; q_n = (\mathrm{Tr}F_n)/n\;,\; G_n = q_n I - F_n \tag{3.6}$$

We shall prove that

$$q_k = b_k, \; k = 1,2,\ldots n \; . \tag{3.7}$$

Also G_n is a null matrix. Thus

$$G_n = 0 \quad \text{and} \quad F_n = b_n I \quad . \tag{3.8}$$

This serves as a check on the operations. Of course, in practice G_n is not null in view of round-off errors. Thus the deviation of the computed G_n from nullity may be taken as a measure of the accumulation of round-off error. If F is nonsingular, then it follows that

$$F^{-1} = \frac{G_{n-1}}{b_n} \quad . \tag{3.9}$$

If F is singular, then G_{n-1} is the matrix adjoint to matrix F. Once the eigenvalues are known, the method of Leverrier produces all the ingredients necessary to compute the corresponding eigenvectors. Let λ_i, $i = 1,2,\ldots n$, be the zeros of $\varphi(\lambda)$. That is, λ_i is an eigenvalue of (3.1). Then the corresponding eigenvector, call it $x^{(i)}$, is proportional to any column of the matrix

$$Q_k = \sum_{r=0}^{n-1} (-)^r \lambda_k^{n-r-1} G_r \; , \; G_o = I \tag{3.10}$$

The proof is as follows. Let

$$\sum_{r=1}^{n} \lambda_r^k = s_k \quad . \tag{3.11}$$

Clearly

$$s_1 = (\mathrm{Tr} F) \quad . \tag{3.12}$$

If λ is an eigenvalue of F, then λ^k is an eigenvalue of F^k. So

$$s_k = (\mathrm{Tr}F^k) \quad . \tag{3.13}$$

Now there is a connection between the s_k's and the b_k's known as Newton's identity [7]. It reads

$$kb_k = (-)^{k-1}\left[s_k - b_1 s_{k-1} + b_2 s_{k-2} + \ldots + (-)^{k-1} b_{k-1} s_1\right] \quad . \tag{3.14}$$

Thus the computational process is reduced to the evaluation of successive powers of the matrix F. Our proof now proceeds by mathematical induction. Obviously $b_1 = (\mathrm{Tr}F) = q_1$. So assume that $q_r = b_r$ for $r = 1,2,\ldots k-1$. We show that $q_k = b_k$. By the algorithm (3.6), we have

$$F_k = (-)^{k-1}\left[F^k - b_1 F^{k-1} + b_2 F^{k-2} + \ldots + (-)^{k-1} b_{k-1} F\right] . \tag{3.15}$$

So

$$\begin{aligned}(\mathrm{Tr}F_k) = kq_k &= (-)^{k-1}\left[(T\ F^k) - b_1(T_r F^{k-1}) + b_2(T_r F^{k-2}) + \ldots + (-)^{k-1} b_{k-1}(\mathrm{Tr}F)\right] \\ &= (-)^{k-1}\left[s_k - b_1 s_{k-1} + b_2 s_{k-2} + \ldots + (-)^{k-1} b_{k-1} s_1\right] \\ &= kb_k \end{aligned} \tag{3.16}$$

in view of (3.12)-(3.14). Thus $q_k = b_k$ and the induction is complete. By the Hamilton-Cayley theorem, a matrix satisfies its own characteristc equation. That is, see (2.3),

$$\sum_{k=0}^{n} (-)^k b_{n-k} F^k = 0 \quad . \tag{3.17}$$

Now put $k = n$ in (3.15) and when this is combined with (3.17), we get

$$F_n = b_n I \quad \text{or} \quad G_n = 0 \quad ,$$

which is the statement (3.8).

From (3.17), we have

$$\sum_{k=1}^{n} (-)^k b_{n-k} F^{k-1} = -b_n F^{-1} \quad . \tag{3.18}$$

If this is compared with (3.15) for $k = n-1$, we get

$$b_n F^{-1} = b_{n-1} I - F_{n-1} = G_{n-1}$$

which is the statement (3.9).

Using (3.10), we have

$$(\lambda_k I - F) Q_k = \sum_{r=0}^{n} (-)^r \lambda_k^{n-r} (F G_{r-1} + G_r) \ , \ G_{-1} = 0 \ ,$$

and with the aid of (3.6), (3.7) and (3.3),

$$(\lambda_k I - F) Q_k = \sum_{r=0}^{n} (-)^r \lambda_k^{n-r} b_r = 0 \quad \text{or} \quad Fu = \lambda_k u \ . \tag{3.19}$$

That is, any column u of the matrix Q_k is the eigenvector corresponding to the eigenvalue λ_k .

We remark that in terms of our original problem, see (1.2), we do not require either the eigenvalues or eigenvectors of F unless s_o is an eigenvalue of the system (1.2). We only require the b_k's . However, for the sake of clarity and completeness, we have presented the method of Leverrier in its entirety.

IV. THE LAGRANGIAN INTERPOLATING POLYNOMIAL AND COMPLETION OF THE SOLUTION

It is useful to review the theoretical procedure deduced thus far. We begin with

$$(As^2+Bs+C)x = 0 \tag{4.1}$$

or equivalently

$$(Is^2+A^{-1}(Bs+C))x = 0 \quad . \tag{4.2}$$

We consider the system

$$Fx = \lambda x \ , \ \lambda = -s^2 \ , \ F = A^{-1}(Bs_o+C) \tag{4.3}$$

so that (4.2) and (4.3) are identical if $s_o = s$. The characteristic equation for (4.3) may be written as

$$\varphi(\lambda) = \sum_{k=0}^{n} (-)^k b_{n-k}\lambda^k \ , \ b_o = 1 \tag{4.4}$$

It is easy to show that the characteristic equation for (4.1) may be expressed as

$$\partial(\lambda) = \sum_{k=0}^{n} (-)^k c_{n-k}\lambda^k \ , \ c_o = 1 \ , \tag{4.5}$$

where c_k is a polynomial in s of degree k . Let

$$c_k(s) = \sum_{r=0}^{k} g_r^{(k)} s^r \quad . \tag{4.6}$$

Clearly

$$c_k(s_o) = b_k \ , \ k = 1,2,\ldots n \quad . \tag{4.7}$$

If the analysis which leads to (4.4) is repeated for (k+1) distinct values of s_o , then the Lagrangian interpolation method can be used to recover $c_k(s)$. Since $k = 1,2,\ldots,n$, in all(n+1) distinct values of s_o are required. The manner of getting the Lagrangian interpolating polynomial follows. Let $f(x)$ be a polynomial in x of degree r . Suppose that $f(x)$ is known at the (r+1) distinct points x_i , $i = 0,1,\ldots, r$. Let $f_r = f(x_r)$. Then

$$f(x) = \sum_{m=0}^{r+1} \frac{A_m(x) f_m}{A_m(x_m)} \quad ,$$

$$A_m(x) = (x-x_o)(x-x_1)\ldots(x-x_{m-1})(x-x_{m+1})\ldots(x-x_r) = \prod_{k=0}^{r}{}' \ (x-x_k) \quad , \tag{4.8}$$

where a $'$ indicates that the factor $(x-x_m)$ is omitted. If

$$g_n(y) = \prod_{i=1}^{n} (y-y_i) = \sum_{k=0}^{n} (-)^k p_k y^{n-k} \ , \ p_o = 0 \quad , \tag{4.9}$$

and

$$S_k = \sum_{r=1}^{k} y_i^k \ , \ S_1 = p_1 \quad , \tag{4.10}$$

then the p_k's are readily evaluated using the recurrence formula

$$p_k = \frac{(-)^{k-1}}{k}\left[S_k-p_1S_{k-1}+p_2S_{k-2}+\ldots+(-)^{k-1}p_{k-1}S_1\right] . \qquad (4.11)$$

The latter is the same as (3.14).

Thus the coefficients $g_r^{(k)}$ are known and the combination (4.5) and (4.6) yields the characteristic equation. The roots of the characteristic equation may be found using Bairstow's iteration procedure, see, for example, [6]. This technique is well known and is described in numerous sources other than the one referenced. We dispense with further details, suffice it to say that it is a generalization of the Newton-Raphson procedure for a single root as it removes quadratic factors from a given polynomial, and so is efficient for the recovery of complex zeros. Once the zeros are known, the corresponding eigenvectors are found by solving linear systems of equations of (4.1) with one equation omitted.

V. RECOMMENDATIONS FOR FUTURE RESEARCH

In this section we list some areas for future research bearing on the problem of finding the eigenvalues and eigenvectors of matrices whose elements are not linear functions of the variable. The present report gives a procedure for solving the problem if the elements are at most quadratic. A natural research problem is the extension of present techniques and development of new procedures to solve the problem when the matrix elements are arbitrary polynomials in the variable. This is important for the applications since it is known that the control equation can introduce polynomials of high order.

As remarked in the main body of this report, the algorithms now employed are not necessarily the most economical from the point of view of machine computation. On the other hand, we noted that the techniques used for developing the characteristic equation of (2.1) may be described as universal in the sense that no divisions are required. Thus, the usual round-off difficulties inherent in methods which require division by pivotal elements are eliminated. Even so, we believe that studies should be made with other procedures for evaluation of the characteristic equation. Another possible economy would arise if the characteristic equation could be recovered by evaluation of the determinant of $D(s)$, see (1.3), for a sufficient number of distinct values of s followed by use of the Lagrangian interpolation formula. This procedure if successful would eliminate need for the method of Leverrier. This aspect should also be investigated.

Often in the analysis of a physical system, one is interested in the effect on stability produced by variation of a parameter. Provided that the parameter change is not too great, it would seem that once the eigenvalues are known for a given state, the corresponding eigenvalues for a slightly changed state could be quickly determined by a perturbation process. Thus, perturbation, and in general iterative techniques for the solution of the general problem, should be investigated.

Another area of interest is the application of root-locus methods on control system design to analyze system stability when certain system parameters are permitted to vary.

REFERENCES

1. Lancaster, P., Arch Rat. Mech. Anal. 8 (1961), 309-322.

2. Faddeeva, V.N., Computational Methods of Linear Algebra, Dover Publications, Inc., New York, 1959.

3. Faddeev, D.K., and Faddeeva, V.N., Computational Methods of Linear Algebra, W.H. Freeman and Co., San Francisco, 1963.

4. Fettis, H.E., Quart. Appl. Math. 8 (1950), 206-212.

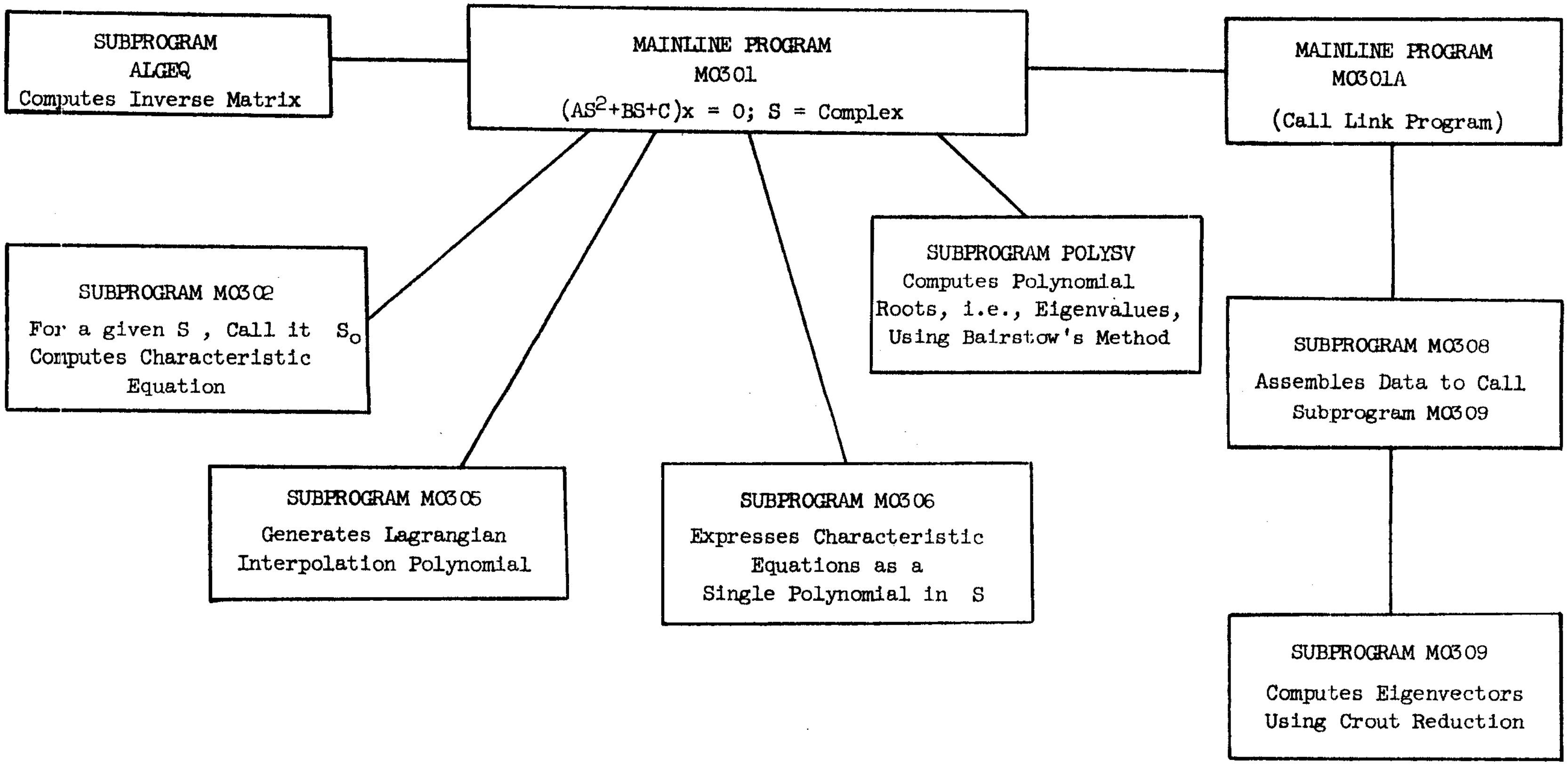
5. Householder, A.S., The Theory of Matrices in Numerical Analysis, Blaisdell Publishing Co., New York, 1964.

6. Hildebrand, F.B., Introduction to Numerical Analysis, McGraw-Hill Book Co., New York, 1956.

7. Bochér, M., Introduction to Higher Algebra, The Macmillan Co., New York, 1936.

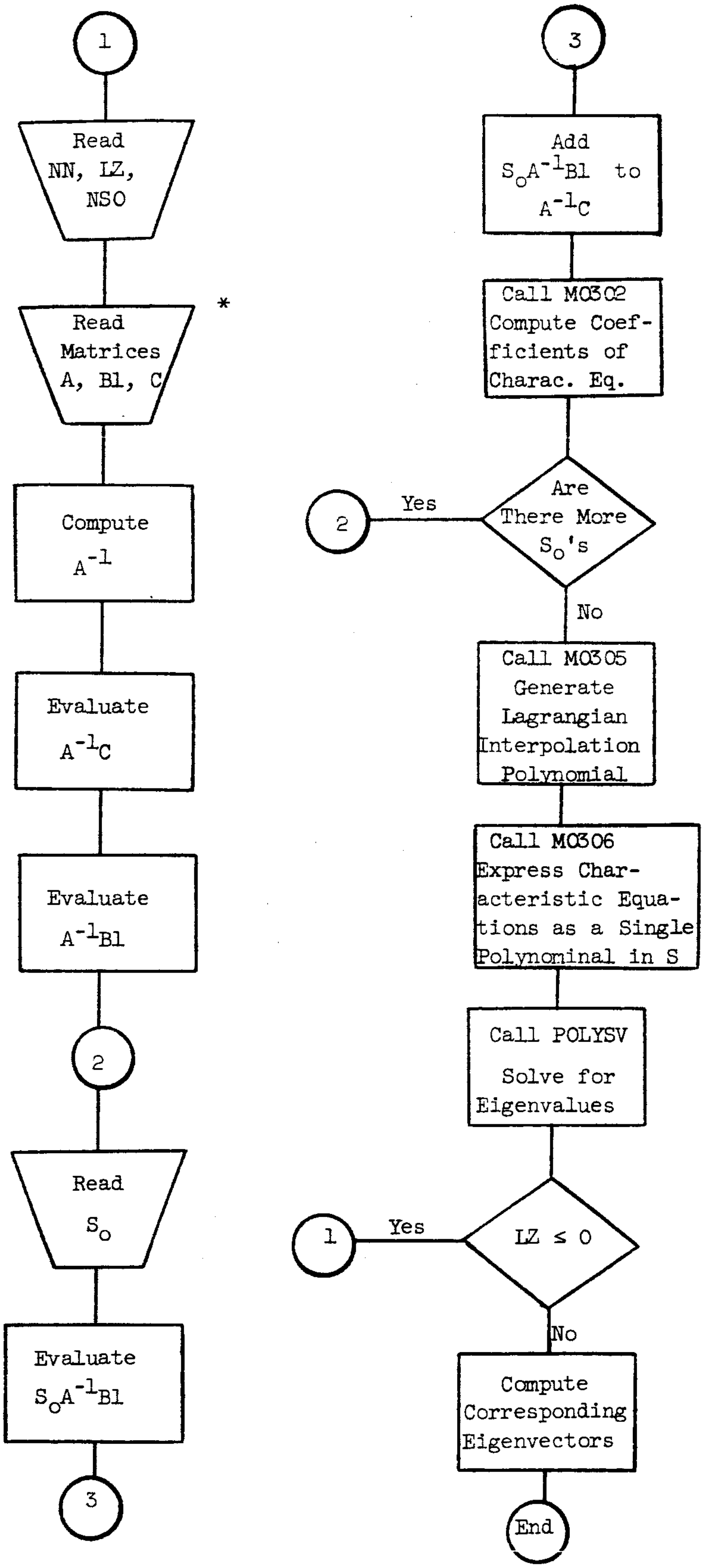
APPENDIX A

In this Appendix we present a flow chart for the evaluation of the eigenvalues and eigenvectors for the system $(AS^2+BS+C)x = 0$ as described in the main body of the report. Also included is a description of the various FORTRAN II-D programs and subroutines written for the IBM 1620 computer.



PROGRAM FLOW CHART

FLOW CHART OF MAINLINE M0301



* The matrix B used in the notation for the general program flow chart is now named B1, as the designator B is used in another connection in subprogram M0305.

PROGRAM M0301

Program M0301 is the mainline program which calls several subprograms and links with mainline program M0301A to compute the eigenvalues and the corresponding eigenvectors of a given matrix whose elements are polynomials of degree no greater than two.

Restrictions or limitations:

1. The input matrices must be square. Also, the matrix whose elements are coefficients of S^2 must be nonsingular. See the note below.

2. The input coefficients of the polynomials must be real numbers.

3. The number of values of S_o should equal the size of the input square matrices plus one.

4. Compatibility of dimensions is necessary between the mainline program and its subprograms and linked programs.

5. SUBROUTINE POLYSV will fail if the polynomial contains a repeated quadratic factor. However, the possibility of this happening is quite rare.

Note: If $A = (a_{ij})$, $B = (b_{ij})$, $C = (c_{ij})$ and $x = (x_1, x_2, ..x_n)$, then the matrix system

$$(AS^2+BS+C)x = 0$$

may be also written as

$$\sum_{j=1}^{n} (a_{ij}S^2+b_{ij}S+c_{ij})x_j = 0 \text{ , } i = 1,2,..n$$

In missile systems, one of the modes at least is a rigid body mode. Suppose x_1 is the rigid body coordinate. Then original data might be given in such a form that $a_{i1} = 0$ for $i = 1,2,...n$. In this event, the raw data must be conditioned before using program M0301 as we require A to be nonsingular. To prepare the data in such a situation, replace x_1 by a new coordinate

y_1/S . Then the coefficient of (y_1/S) has the form $(b_{i1}S^2+c_{i1}S)$. Other rigid body modes, if present, are treated in a similar fashion. The conditioned data are now in a suitable form to get the eigenvalues using the above program. However, once an eigenvalue S is known, subroutines M0308 and M0309 produce the corresponding vector x . Thus the input data used in the latter routines are the original data, not the conditioned data employed to get the eigenvalue.

MAINLINE PROGRAM M0301

Input

READ STATEMENT: 100 READ 200, NN, LZ, NSO

FORMAT STATEMENT: 200 FORMAT (3I2)

Card Columns	Data	Definition of Data
1-3	NN	Size of input square matrix
4-6	LZ	LZ is a control code. If $LZ > 0$, program computes eigenvectors corresponding to each eigenvalue. If $LZ \leq 0$, program computes eigenvalues but no eigenvectors.
7-9	NSO	Number of values for S_o to be used

NN, LZ, NSO are fixed point data; right justify these data in their respective card columns.

READ STATEMENT: READ 201, ((A(I,J), I = 1, NN), J = 1, NN)

FORMAT STATEMENT: 201 FORMAT (3 F 25.0)

Card Columns	Data	Definition of Data
1-25 26-50 51-75	A(I,J)	A is a matrix whose elements are coefficients of S^2; read first column of data, then the second column, etc.

Example of Data on Cards When A is a 4 x 4 Matrix

	Card Columns 1-25	26-50	51-75
1st Card	A(1,1)	A(2,1)	A(3,1)
2nd Card	A(4,1)	A(1,2)	A(2,2)
3rd Card	A(3,2) etc.	A(4,2)	A(1,3)

READ STATEMENT: READ 201, ((B1(I,J), I = 1,NN), J = 1,NN)

FORMAT STATEMENT: 201 FORMAT (3 F 25.0)

Card Columns	Data	Definition of Data
1-25 26-50 51-75	B1(1,1)	B1 = matrix whose elements are coefficients of S ; read 1st column first, then 2nd column, etc.

If the input matrix B1 is a 4 x 4 matrix

	Card Columns 1-25	26-50	51-75
1st Card	B1(1,1)	B1(2,1)	B1(3,1)
2nd Card	B1(4,1)	B1(1,2)	B1(2,2)
3rd Card	B1(3,2)	B1(4,2)	B1(1,3) etc.

READ STATEMENT: READ 201 ((C(I,J), I = 1, NN), J = 1, NN)

FORMAT STATEMENT: 201 FORMAT (3 F 25.0)

Card Columns	Data	Definition of Data
1-25 26-50 51-75	C(I,J)	C = matrix whose elements are independent of S . Read 1st column first, then 2nd column, etc.

If the input matrix = C(I,J), I = J = 4 , the data cards should be read:

Card Columns	1-25	26-50	51-75
1st Card	C(1,1)	C(2,1)	C(3,1)
2nd Card	C(4,1)	C(1,2)	C(2,2)
3rd Card	C(3,2)	C(4,2)	C(1,3) etc.

READ STATEMENT: 126 READ 202, SS

FORMAT STATEMENT: 202 FORMAT (F 25.0)

Card Columns	Data	Definition of Data
1-25	SS	SS = S_o . Note: There should be as many cards containing values of SS as the value of NSO in the first card of the input data deck.

SUBROUTINE ALGEQ

This subroutine solves for the inverse of a matrix using a modified Gaussian method.

SUBROUTINE ALGEQ (A, B1, NA, MA, DET)

Definitions of the variables in the argument list

A(I,J), I, J = 1,2,...NA;= coefficient matrix

B1(I,J), I,J = 1,2,...NA;= initially this is a unit matrix; after execution of subroutine ALGEQ,this contains the inverse of the coefficient matrix A(I,J)

NA = dimension of the square matrix A(I,J)

MA = 1

DET = determinant of A(I,J)

SUBROUTINE M0302

This subroutine is used to solve for the coefficients of the characteristic equation corresponding to each value of S_o.

SUBROUTINE M0302 (A, AA, BB, NN, M, MA)

Definitions of the variables in the argument list

NN = dimension of the square matrix

MA = 1,2,..., NN+1

M = 1,2,..., NN

A(I,J), I, J = 1,2,...,NN; = coefficient matrix formed in the mainline program

AA(I,J), I, J = 1,2,...,NN; = name of the intermediate matrices used in determining the coefficients of the characteristic equation

BB(I,J), I = 1,2,...NN+1, J = 1,2,...,NN; = coefficients of the characteristic equation of the form

$$\lambda^n - b_1\lambda^{n-1} + b_2\lambda^{n-2} + \ldots + (-)^n b_n = 0 \quad .$$

SUBPROGRAM M0305

This subroutine uses the Lagrangian interpolation method to interpolate between the coefficients of the characteristic equations corresponding to the various S_o's . The results will be the $g_i^{(j)}$'s as shown below:

$$S^{2n} + S^{2n-2}\left[g_1^{(1)}S_o+g_o^{(1)}\right] + S^{2n-4}\left[g_2^{(2)}S_o^2+g_1^{(2)}S_o+g_o^{(2)}\right]$$

$$+ \ldots + S^2\left[g_{n-1}^{(n-1)}S_o^{n-1}+g_{n-2}^{(n-1)}S_o^{n-2}+\ldots+g_1^{(n-1)}S_o+g_o^{(n-1)}\right]$$

$$+\left[g_n^{(n)}S_o^n+g_{n-1}^{(n)}S_o^{n-1}+g_{n-2}^{(n)}S_o^{n-2}+\ldots+g_1^{(n)}S_o+g_o^{(n)}\right]$$

SUBROUTINE M0305 (B, NX, SO, G, NO, JB)

Definitions of the variables in the argument list

B(I,J), I = 1,2,...NSO, J = 1,2,...NSO-1;= coefficients of the characteristic equations (corresponding to each S_o) computed in subprogram M0302

NX = NN + 1

SO(I), I = 1,2,...NSO;= S_o values

G(J,I), J = 1,2,...NSO-1, I = 1,2,...NSO; = interpolated coefficients $g_i^{(j)}$

NO = NN + 1

JB = NN

SUBPROGRAM M0306

This subprogram reads the output coefficients ($g_i^{(j)}$'s) previously computed in subprogram M0305, sets $S_o = S$, and combines like terms to form the desired characteristic equation.

SUBROUTINE M0306 (NN, G1, NO, JB, M2, PN)

Definitions of the variables in the argument list

NN = dimension of the input square matrices

G1(J,I), J = 1,2,...NSO-1, I = 1,2,...NSO; = interpolated coefficients ($g_i^{(j)}$'s) computed in subprogram M0305

NO = NN + 1

JB = NN

M2 = NN + NN

PN(I) where I = 1,2,...,M2+1;= coefficients of the desired characteristic equation

SUBROUTINE POLYSV

This subprogram uses Bairstow's method to calculate the roots of the characteristic polynomial. Zero roots are automatically removed. The program will fail if the polynomial contains a repeated quadratic factor.

SUBROUTINE POLYSV (N, PN, ZRR, ZRB, ZRC, JR, JC)

Definitions of the variables in the argument list

N = order of the characteristic polynomial

PN(I), I = 1,2,...N+1; = coefficients of the characteristic polynomial

ZRR(I), I = 1,2,...JR; = real roots

ZRB(I), I = 1,2,...JC; = real parts of the complex roots

ZRC(I), I = 1,2,...JC; = imaginary parts of the complex roots

JR = number of real roots

JC = number of complex roots

Definitions of the control values in subprogram POLYSV

ERRD = the number which controls the accuracy to which the coefficients of the quadratic factors are found. When the number used is 10^{-n}, the i^{th} iterates for the coefficients of the quadratic factors are accepted only when they agree with the i-1st iterates to n digits.

TST3D = control number which prevents overflow due to multiplication during iteration for the coefficients of the quadratic factors.

TST2D = number which controls the number of significant digits obtained from the square root routine and is set equal to 10^k. If A_{i-1} and A_i are two consecutive iterates of the square root of A^2 and

$$\left| 1 - \left| \frac{A_{i-1}}{A_i} \right| \right| \le \frac{1}{10^k}$$

is satisfied, then A_i is accepted as the square root of A^2 and is correct up to the k^{th} digit.

TST1D = number which sets the decimal point of the iterates for the coefficients of the quadratic factors to the left of the digits. This is accomplished by successive multiplication of the iterates by 0.1. The value of this number is always 0.1.

Scale = scale factor, here we used 1.0.

Additional input (optional): If it is desired to use different values (other than those the program assigns) for the coefficients of the trial quadratic, $f(s) = s^2 + b_n s + c_n$, enter b_n , c_n and n where n is a quadratic code number.

Sense switch settings: Set all sense switches off. If the optional input is used, set sense switch one on.

Sense switch two on instructs the computer to print the coefficients of the computed quadratic factors.

Output: The eigenvalues of the characteristic polynomial are printed. The real eigenvalues appear first. Each printed eigenvalue is of the form x_i or x_j, y_j where x_i is a real root and x_j , y_j corresponds to the complex eigenvalue $x_j \pm iy_j$.

PROGRAM M0301A

This program links with program M0301 and becomes the mainline program to call subprograms in order to compute the eigenvectors corresponding to each of the eigenvalues computed in program M0301.

Program M0301A is linked to M0301 through this common statement:

COMMON NN, JR, JC, ZRR, ZRB, ZRC

Definitions of the variables in common

NN = dimension of input matrix

JR = number of eigenvalues which are real numbers

JC = number of eigenvalues which are complex numbers

ZRR(I), I = 1,2,...JR; = real eigenvalues

ZRB(I), I = 1,2,...JC; = real parts of the complex eigenvalues

ZRC(I), I = 1,2,...JC; = imaginary parts of the complex eigenvalues

<u>Output</u>: The eigenvectors corresponding to each eigenvalue is printed in the form

$$\begin{array}{ccc} 1 & x_i^{(1)} & y_i^{(1)} \\ 2 & x_i^{(2)} & y_i^{(2)} \\ 3 & x_i^{(3)} & y_i^{(3)} \\ \vdots & \vdots & \vdots \end{array}$$

where the x_i's are the real parts and the y_i's are the imaginary parts. If the eigenvalue is real the y_i's will all be printed as zero.

<u>Input Data for Program M0301A</u>

READ STATEMENTS READ 201, ((A1(I,J), I = 1, NU), J = 1, NU)
READ 201, ((B1(I,J), I = 1, NU), J = 1, NU)
READ 201, ((C(I,J), I = 1, NU), J = 1, NU)

A1 = matrix whose elements are coefficients of S^2

B1 = matrix whose elements are coefficients of S

C = matrix whose elements are coefficents independent of S

NU = NN = dimension of square matrix

Read these data exactly as the initial matrix data were read. Matrices A1, B1, and C are coefficients of the given quadratic polynomial elements.

Note: See the note following restriction 4 in the description of Program M0301.

SUBPROGRAM M0308

This subprogram assembles data for execution of subprogram M0309, and calls subprogram M0309 which solves for the eigenvectors corresponding to each of the eigenvalues.

SUBPROGRAM M0308 (SR, SI, NU, A1, B1, C)

Definitions of the variables in the argument list

SR = real part of a real or complex eigenvalue

SI = imaginary part of a complex eigenvalue

NU = dimension of the input square matrices

A1(I,J), I,J = 1,2,...NU; matrix whose elements are coefficients of S^2 in the given quadratic polynomials.

B1(I,J), I,J = 1,2,...NU; matrix whose elements are coefficients of S in the given quadratic polynomials.

C(I,J), I,J = 1,2,...NU; matrix whose elements are constants in the given quadratic polynomials.

SUBPROGRAM M0309

This subprogram utilizes the Crout reduction method to obtain the eigenvectors corresponding to each of the eigenvalues.

SUBPROGRAM M0309 (AR, AC, C1, CC, XR, XI, NA, DETR, DETI)

Definitions of the variables in the argument list

AR(I,J), I,J = 1,2,...NU-1; real coefficients of a matrix

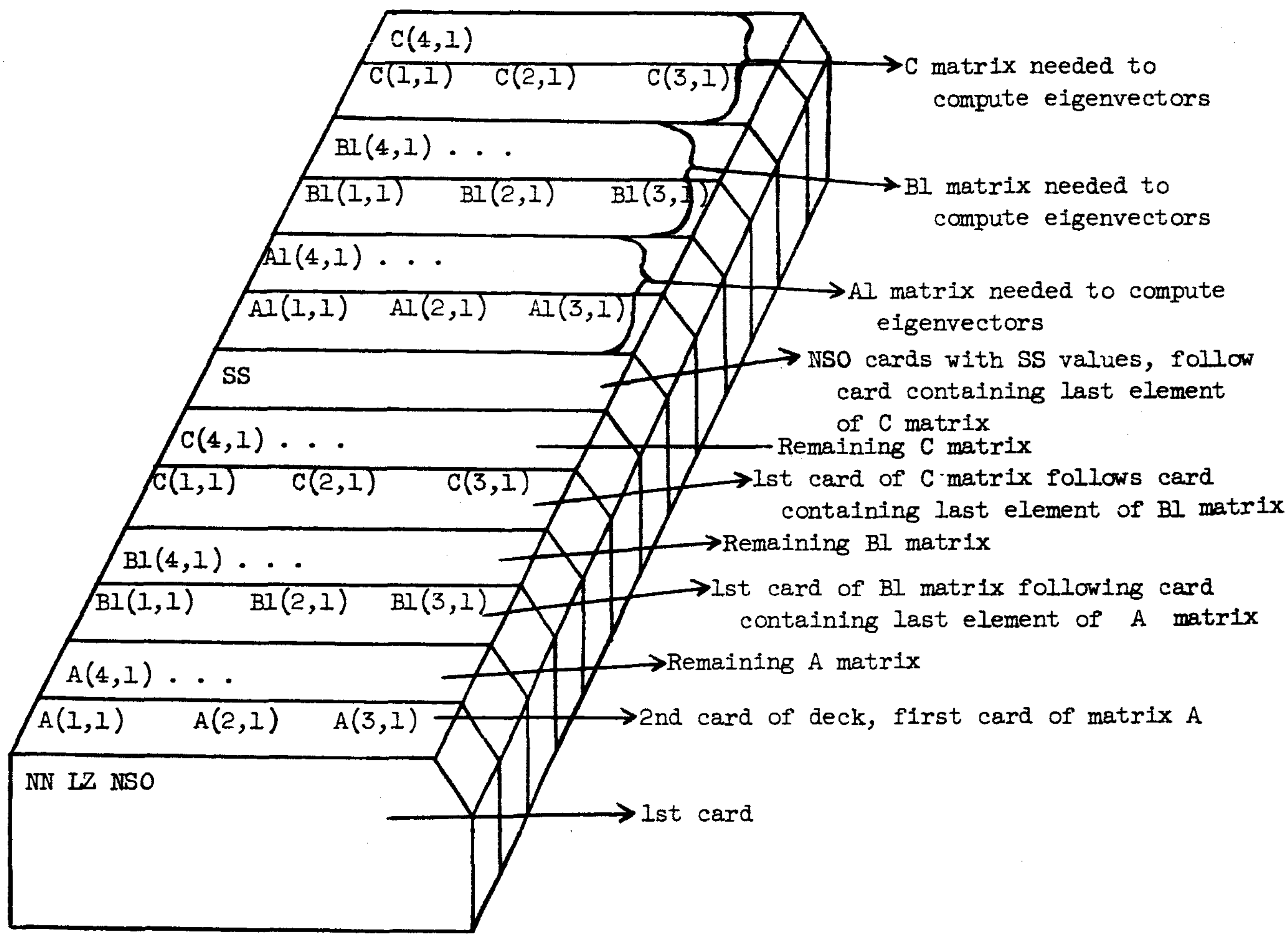
C1(I), I = 1,2,...NU-1; real parts of the coefficients on the right side of the equation $Ax = c$

CC(I), I = 1,2,...NU-1; imaginary parts of the coefficients on the right side of the equation $Ax = c$

XR(I), I = 1,2,...NU; real parts of the entries in the solution eigenvectors

XI(I), I = 1,2,...NU; imaginary parts of the entries in the solution eigenvectors

PHYSICAL APPEARANCE OF INPUT DATA DECK



APPENDIX B

In the sequel, we list the FORTRAN II-D program described in Appendix A.

```
C     PROGRAM M0301
C     COMPUTES THE EIGENVALUES AND EIGENVECTORS FOR
C     (AS2+BS+C)X=0,WHEN S IS COMPLEX
      DIMENSION A(4,4),A1(4,4),B1(4,4),C(4,4),AA(4,4),AC(4,4),AB(4,4),
     1ABSO(4,4),BB(5,4),SO(5),G(4,5),PN(9),ZRR(40),ZRB(20),ZRC(20)
      COMMON NN,JR,JC,ZRR,ZRB,ZRC
  100 READ 200,NN,LZ,NSO
      NN=ABSF(NN)
C     A(NN,NN) = MATRIX WHOSE ELEMENTS ARE COEFFICIENTS OF S SQUARED,
C     B1(NN,NN) = MATRIX WHOSE ELEMENTS ARE COEFFICIENTS OF S,
C     C(NN,NN) = MATRIX WHOSE ELEMENTS ARE CONSTANTS.
      READ 201,((A(I,J),I=1,NN),J=1,NN)
      READ 201,((B1(I,J),I=1,NN),J=1,NN)
      READ 201,((C(I,J),I=1,NN),J=1,NN)
      DO 101 I=1,NN
      DO 101 J=1,NN
      IF(I-J)103,102,103
  102 AA(I,J)=1.0
      GO TO 101
  103 AA(I,J)=0.0
  101 A1(I,J)=A(I,J)
C     COMPUTES INVERSE MATRIX (MODIFIED GAUSSIAN METHOD)
C     A(I,J) X X(J)=AA(I,J)
C     A=COEFFICIENT MATRIX,AA=RIGHT HAND SIDE VECTOR OR MATRIX,NN IS
C     SIZE OF THE SQUARE MATRIX A,NN IS 1 IF AA IS A VECTOR,
C     DET=VALUE OF DETERMINANT A-MATRIX,AFTER EXECUTION
      CALL ALGEQ(A,AA,NN,NN,DET)
      DO 104 I=1,NN
      DO 104 J=1,NN
      AC(I,J)=0.0
      AB(I,J)=0.0
      DO 104 K=1,NN
      AC(I,J)=AA(I,K)*C(K,J)+AC(I,J)
  104 AB(I,J)=AA(I,K)*B1(K,J)+AB(I,J)
      MA=0
  106 READ 202,SS
      MA=MA+1
      SO(MA)=SS
      DO 105 I=1,NN
      DO 105 J=1,NN
  105 ABSO(I,J)=SS*AB(I,J)
      DO 107 I=1,NN
      DO 107 J=1,NN
      A(I,J)=0.0
      AA(I,J)=0.0
      A(I,J)=ABSO(I,J)+AC(I,J)
  107 AA(I,J)=A(I,J)
      M=0
C     OBTAINS COEFFICIENTS OF THE CHARACTERISTIC EQUATION
  122 CALL M0302(A,AA,BB,NN,M,MA)
      IF(NSO-MA)123,123,106
C     LAGRANGIAN INTERPOLATION
  123 CALL M0305(BB,MA,SO,G,NO,JB)
C     COMBINES LIKE TERMS ,S=S NAUGHT
      CALL M0306(NN,G,NO,JB,M2,PN)
```

```
C     COMPUTES POLYNOMIAL ROOTS (BAIRSTOW,S METHOD)
      CALL POLYSV(M2,PN,ZRR,ZRB,ZRC,JR,JC)
      IF(LZ)100,100,149
C     IF LZ IS GREATER THAN ZERO COMPUTE EIGENVECTORS
  149 CALL LINK(M0301A)
  200 FORMAT(3I2)
  201 FORMAT(3F25.0)
  202 FORMAT(F25.0)
      END


      SUBROUTINE M0302(A,AA,BB,NN,M,MA)
C     SUBROUTINE USED IN OBTAINING THE COEFFICIENTS OF THE
C     CHARACTERISTIC EQUATION
      DIMENSION A(4,4),AA(4,4),ASQ(4,4),BB(5,4)
  101 M=M+1
      BB(MA,M)=0.0
      DO 102 I=1,NN
  102 BB(MA,M)=AA(I,I)+BB(MA,M)
      DIV=M
      BB(MA,M)=BB(MA,M)/DIV
  107 DO 103 I=1,NN
      DO 103 J=1,NN
      ASQ(I,J)=0.0
      DO 103 K=1,NN
  103 ASQ(I,J)=A(I,K)*AA(K,J)+ASQ(I,J)
      DO 104 I=1,NN
      DO 104 J=1,NN
  104 AA(I,J)=(BB(MA,M)*A(I,J))-ASQ(I,J)
  106 IF(M-NN)101,105,105
  105 RETURN
      END


      SUBROUTINE ALGEQ(A,B1,NA,MA,DET)
C     FIND SOLUTION TO A*X=B WHERE A IS AN NA BY NA MATRIX
      DIMENSION A(4,4),B1(4,4)
      DET=1.
      DO 2100 J=1,NA
      X=0.
      K=0
      DO 2200 I=J,NA
      IF(ABSF(A(I,J))-X)2200,2200,2150
 2150 X=ABSF(A(I,J))
      K=I
 2200 CONTINUE
      IF(K-J)2900,2290,2250
 2250 DO 2260 I=J,NA
      X=A(J,I)
      A(J,I)=A(K,I)
 2260 A(K,I)=X
      DO 2280 I=1,MA
      X=B1(J,I)
      B1(J,I)=B1(K,I)
 2280 B1(K,I)=X
```

```
      DET=-DET
 2290 X=A(J,J)
      DO 2300 I=J,NA
 2300 A(J,I)=A(J,I)/X
      DO 2310 I=1,MA
 2310 B1(J,I)=B1(J,I)/X
      DET=DET*X
      IF(J-NA)2320,2400,2400
 2320 L=J+1
      DO 2100 I=L,NA
      X=A(I,J)
      DO 2340 K=L,NA
 2340 A(I,K)=A(I,K)-X*A(J,K)
      DO 2100 K=1,MA
 2100 B1(I,K)=B1(I,K)-X*B1(J,K)
 2400 DO 2600 KK=1,MA
      DO 2600 I=2,NA
      K=NA+1-I
      J=K+1
      DO 2600 L=J,NA
 2600 B1(K,KK)=B1(K,KK)-A(K,L)*B1(L,KK)
      GO TO 2950
 2900 DET=0.
      PRINT 2999
 2999 FORMAT(30H A MATRIX IN ALGEQ IS SINGULAR)
 2950 RETURN
      END


      SUBROUTINE M0306(NN,G1,NO,JB,M2,PN)
C     FOR CASE S NAUGHT = S
C     COMBINES LIKE TERMS TO FORM THE CHARACTERISTIC EQUATION
      DIMENSION G1(4,5),G2(4,5),PN(9)
  320 DO 321 K=1,JB
      DO 321 J=1,NO
  321 G2(K,J)=0.0
      DO 322 K=1,JB
      K1=K+1
      DO 322 J=1,K1
      K2=K1-J+1
  322 G2(K,J)=G1(K,K2)
      POL=0.0
      M2=2*NN
      PN(M2+1)=1.0
      DO 300 K=1,M2
      M1=M2-K+1
      IF(M1)310,310,311
  311 IF(K-NO)301,301,302
  302 L=L+1
      LL=0
      GO TO 303
  301 KK=K
      L=1
  303 DO 304 J=L,KK
      IF(J-JB)309,309,306
```

```
  309 IF(L-1)305,305,308
  308 LL=LL+1
      K1=KK-LL+1
      GO TO 304
  305 K1=KK-J+1
  304 POL=G2(J,K1)+POL
  306 PN(M1)=POL
  300 POL=0.0
C     PN(K),S ARE THE COEFFICIENTS OF THE CHARACTERISTIC EQUATION
C     PN(1) IS THE CONSTANT TERM
      IF(ABSF(PN(1))-.1E-08)323,323,310
  323 DO 324 K=1,M2
      PN(K)=PN(K+1)
  324 CONTINUE
      M2=M2-1
      RETURN
  310 M4=M2+1
      RETURN
      END


      SUBROUTINE POLYSV(N,PN,ZRR,ZRB,ZRC,JR,JC)
C     ZEROS OF POLYNOMIALS
      DIMENSION PN(9),B(40),G(40),ZRB(20),ZRC(20),ZRR(40)
      NQ=0
      ERRD=1.E-13
      TST3D=1.E+06
      TST2D=1.E+15
      TST1D=.1
      SCALE=1.
      IF(SENSE SWITCH 3)101,100
 101  READ 999,ERRD,TST3D,TST2D,TST1D,SCALE
 100  LP=N+1
      ZPLP=PN(LP)
      DELT=1./PN(LP)
      DO 102 J=1,LP
 102  PN(J)=PN(J)*DELT
      JR=1
      JC=1
 57   IF(PN(1))2,58,2
 58   LP=N
      N=N-1
      ZRR(JR)=.0
      JR=JR+1
      DO 59 J=1,LP
 59   PN(J)=PN(J+1)
      PN(LP+1)=.0
      GO TO 57
 2    B(N+1)=.0
      B(N)=.0
      B(N-1)=1.
      G(N-1)=.0
      G(N-2)=.0
      CA=.0
      DA=.0
```

```
      IF(SENSE SWITCH 1)20,21
20    READ 996,CA,DA
21    RUB=.0
      DB=.0
      NQ=NQ+1
3     NP=N-1
      DO 4 K=2,NP
      M=N-K
4     B(M)=PN(M+2)-B(M+2)*DA-B(M+1)*CA
      IF(N-3)17,7,5
5     DO 6 K=3,NP
      M=N-K
6     G(M)=B(M+2)-G(M+1)*CA-G(M+2)*DA
7     GA=B(2)-G(1)*CA-G(2)*DA
      GB=B(1)-GA*CA-G(1)*DA
      R1=PN(2)-B(2)*DA-B(1)*CA
      RO=PN(1)-B(1)*DA
      IF(GA-GB)120,121,121
120   XX=GB
      GO TO 122
121   XX=GA
122   IF(TST3D-XX)123,140,140
123   XX=1./XX
      GO TO 150
140   XX=1.
150   DELTA=GB*XX*(B(1)-G(1)*DA)+GA*XX*GA*DA
      IF(DELTA)45,44,45
44    DELTA=1.
45    CA=CA+((B(1)-G(1)*DA)*XX*R1-GA*XX*RO)/DELTA
      DA=DA+(GA*XX*R1*DA+GB*XX*RO)/DELTA
      ERRB=DA-DB
      ERRA=CA-RUB
      RUB=CA
      DB=DA
      IF(ERRA)60,61,61
60    ERRA=-ERRA
61    IF(ERRB)62,50,50
62    ERRB=-ERRB
50    IF(1.-ERRA)54,55,55
54    ERRA=ERRA*TST1D
      GO TO 50
55    IF(1.-ERRB)56,23,23
56    ERRB=ERRB*TST1D
      GO TO 55
23    IF(ERRD-ERRA)3,8,8
8     IF(ERRD-ERRB)3,9,9
9     IF(SENSE SWITCH 2)24,25
24    PRINT 995,NQ,CA,DA
25    REST=-CA*.5
      SURD=CA*CA-DA*4.
      IF(SURD)10,11,12
10    BURD=-SURD
70    X1=1.
71    X2=.5*(X1+BURD/X1)
      TEST=TST2D*(X2-X1)/X2
```

```
      IF(TEST)72,73,73
72    TEST=-TEST
73    IF(1.-TEST)74,75,75
74    X1=X2
      GO TO 71
75    BURD=X2
      IF(SURD)81,82,82
81    SURD=BURD*.5
      ZRB(JC)=REST
      ZRC(JC)=SURD
      JC=JC+1
      GO TO 13
11    ZRR(JR)=REST
      ZRR(JR+1)=REST
      JR=JR+2
      GO TO 13
12    BURD=SURD
      GO TO 70
82    SURD=BURD*.5
      ZRR(JR)=REST+SURD
      ZRR(JR+1)=REST-SURD
      JR=JR+2
13    N=N-2
      IF(N-1)17,16,14
14    IF(N-2)15,15,18
15    CA=B(2)
      DA=B(1)
      GO TO 9
18    NP=N+1
      DO 19 I=1,NP
19    PN(I)=B(I)
      GO TO 2
16    S=-B(1)/B(2)
      ZRR(JR)=S
      JR=JR+1
17    JC=JC-1
      JR=JR-1
      PRINT 993
      IF(JR)106,106,105
105   DO 103 J=1,JR
      ZRR(J)=ZRR(J)*SCALE
      IF(SENSE SWITCH 4)109,103
109   PUNCH 994,ZRR(J)
103   PRINT 994,ZRR(J)
106   IF(JC)108,108,107
107   DO 104 J=1,JC
      ZRB(J)=ZRB(J)*SCALE
      ZRC(J)=ZRC(J)*SCALE
      IF(SENSE SWITCH 4)110,104
110   PUNCH 994,ZRB(J),ZRC(J)
104   PRINT 994,ZRB(J),ZRC(J)
108   RETURN
993   FORMAT(//16H1THE EIGENVALUES)
994   FORMAT(/2F30.20)
995   FORMAT(/15H0FOR QUADRATIC I3,/10H    S**2 + F11.4,5H S + F11.4)
```

```
  996   FORMAT(2F9.0)
  999   FORMAT(3E9.0,2F5.1)
        END


        SUBROUTINE M0305(B,NX,SO,G,NO,JB)
C       INTERPOLATION METHOD
        DIMENSION B(5,4),SO(5),G(4,5),S(5),DEN(5),P(5)
  100   NY=NX-1
        G(1,1)=((B(1,1)*(-SO(2)))/(SO(1)-SO(2)))+((B(2,1)*(-SO(1)))/
       1(SO(2)-SO(1)))
        G(1,2)=(B(1,1)/(SO(1)-SO(2)))+(B(2,1)/(SO(2)-SO(1)))
        DO 112 I=2,NY
        ND=I+1
        DO 112 J=1,ND
  112   G(I,J)=0.0
        NO=2
  101   NO=NO+1
        DO 110 JA=1,NO
        DEN(JA)=1.0
        DO 106 J=1,NO
  106   S(J)=0.0
        IX=0
  107   IX=IX+1
        DO 103 J=1,NO
        IF(SO(J))120,103,120
  120   IF(JA-J)108,103,108
  108   S(IX)=SO(J)**IX+S(IX)
  103   CONTINUE
        IF(IX-NO)107,109,109
  109   DO 102 J=1,NO
        IF(JA-J)116,102,116
  116   DEN(JA)=(SO(JA)-SO(J))*DEN(JA)
  102   CONTINUE
        P(1)=S(1)
        JB=NO-1
        DO 105 K=2,JB
        XK=K
        P(K)=0.0
        DO 104 J=2,K
        K1=K-J+1
        JJ1=J-1
  104   P(K)=((-1.)**JJ1)*(P(J-1)*S(K1))+P(K)
        KK1=K-1
        P(K)=((-1.)**KK1)*(1./XK)*(S(K)+P(K))
  105   CONTINUE
        DO 110 K=1,NO
        NB=NO-K+1
        KK=K-1
        IF(NB-NO)114,113,114
  113   G(JB,NO)=(((-1.)**KK)*B(JA,JB))/DEN(JA)+G(JB,NO)
        GO TO 110
  114   G(JB,NB)=(((-1.)**KK)*P(KK)*B(JA,JB))/DEN(JA)+G(JB,NB)
  110   CONTINUE
  115   IF(NO-NY)101,101,117
```

```
  117 CONTINUE
      RETURN
      END

      SUBROUTINE M0308(SR,SI,NU,A1,B1,C)
C     SOLVES THE SYSTEM (AS2+BS+C)X=0,OR EX=0,FOR COMPLEX NUMBERS
      DIMENSION A1(4,4),B1(4,4),C(4,4),E(4,4),EC(4,4),C1(4),CC(4),XR(4),
     1XI(4),AR(3,3),AC(3,3)
      SR2=SR*SR-SI*SI
      SI2=SR*SI+SI*SR
      DO 102 J=1,NU
      DO 102 I=1,NU
      E(I,J)=A1(I,J)*SR2+B1(I,J)*SR+C(I,J)
  102 EC(I,J)=A1(I,J)*SI2+B1(I,J)*SI
C     FIND SMALLEST ELEMENT
      IF(SI)161,162,161
  162 DO 163 K=1,NU
  163 XR(K)=E(K,K)
      GO TO 164
  161 DO 160 K=1,NU
      ARG=(E(K,K)*E(K,K)+EC(K,K)*EC(K,K))
  160 XR(K)=SQRTF(ARG)
  164 K1=1
  105 TEST=XR(K1)
      DO 111 J=1,NU
      IF(J-K1)142,111,142
  142 IF(XR(J))143,144,145
  143 IF(TEST)146,150,150
  146 IF(TEST-XR(J))111,111,150
  144 IF(TEST)111,111,150
  145 IF(TEST)111,111,147
  147 IF(TEST-XR(J))111,111,150
  150 K1=J
      GO TO 105
  111 CONTINUE
C     NEGLECT K-TH EQUATION
      NW=NU-1
      DO 115 I=1,NW
      IK=I
      IF(K1-I)117,117,116
  117 IK=I+1
  116 DO 115 J=1,NW
      JK=J
      IF(K1-J)119,119,118
  119 JK=J+1
  118 AR(I,J)=E(IK,JK)
      AC(I,J)=EC(IK,JK)
  115 CONTINUE
      DO 120 K=1,NW
      KA=K
      IF(K1-K)121,121,165
  121 KA=K+1
  165 C1(K)=-E(KA,K1)
  120 CC(K)=-EC(KA,K1)
```

```
      CALL M0309(AR,AC,C1,CC,XR,XI,NW)
      PRINT 203
      DO 128 M=1,NU
      IF(K1-M)129,130,131
  130 C1(M)=1.0
      CC(M)=0.0
      GO TO 128
  129 C1(M)=XR(M-1)
      CC(M)=XI(M-1)
      GO TO 128
  131 C1(M)=XR(M)
      CC(M)=XI(M)
  128 CONTINUE
      DO 133 K=1,NU
  133 PRINT 206,K,C1(K),CC(K)
      RETURN
  203 FORMAT(30H0THE CORRESPONDING EIGENVECTOR//)
  206 FORMAT(I2,2E25.16)
      END


C     PROGRAM M0301A
C     COMPUTES EIGENVECTORS
      DIMENSION ZRR(40),ZRB(20),ZRC(20),ZRI(10),A1(4,4),B1(4,4),C(4,4)
      COMMON NN,JR,JC,ZRR,ZRB,ZRC
      READ 201,((A1(I,J),I=1,NN),J=1,NN)
      READ 201,((B1(I,J),I=1,NN),J=1,NN)
      READ 201,((C(I,J),I=1,NN),J=1,NN)
  149 NUM=0
  140 IF(JR)137,137,141
  141 DO 130 K=1,JR
  130 ZRI(K)=0.0
  152 NUM=NUM+1
      PRINT 203,ZRR(NUM),ZRI(NUM)
      CALL M0308(ZRR(NUM),ZRI(NUM),NN,A1,B1,C)
      IF(JR-NUM)137,137,152
  137 IF(JC)126,126,133
  133 K1=JR+1
      DO 135 J=1,JC
      ZRR(K1)=ZRB(J)
      ZRI(K1)=ZRC(J)
  135 K1=K1+1
      KE=K1-1
  153 NUM=NUM+1
      PRINT 203,ZRR(NUM),ZRI(NUM)
      CALL M0308(ZRR(NUM),ZRI(NUM),NN,A1,B1,C)
      IF(KE-NUM)126,126,153
  126 CALL EXIT
  201 FORMAT(3F25.0)
  203 FORMAT(17H1THE EIGENVALUE =2E24.15//)
      END


      SUBROUTINE M0309(AR,AC,C1,CC,XR,XI,NA)
C     CROUT REDUCTION
```

```
      DIMENSION AR(3,3),AC(3,3),AP(3,3),ACP(3,3),C1(4),CC(4),CCP(4),
     1CP(4),XR(4),XI(4)
  100 DO 101 J=1,NA
      K=0
      DO 106 I=1,NA
      IF(ABSF(AR(I,J)))103,102,103
  102 IF(ABSF(AC(I,J)))103,106,103
  103 K=I
  106 CONTINUE
      IF(K-1)130,104,104
  104 DO 117 I=J,NA
      SAM=0.0
      SUM=0.0
      IF(1-J)107,108,107
  107 JN=J-1
      DO 109 K=1,JN
      SAM=SAM+(AP(I,K)*ACP(K,J)+ACP(I,K)*AP(K,J))
  109 SUM=SUM+(AP(I,K)*AP(K,J)-ACP(I,K)*ACP(K,J))
  108 ACP(I,J)=AC(I,J)-SAM
      AP(I,J)=AR(I,J)-SUM
  117 CONTINUE
      IF(NA-J)120,125,113
  113 I=J
      KK=J
  114 KK=KK+1
      SAM=0.0
      SUM=0.0
  105 IF(1-I)110,111,110
  110 JM=I-1
      DO 112 K=1,JM
      SAM=SAM+(AP(I,K)*ACP(K,KK)+ACP(I,K)*AP(K,KK))
  112 SUM=SUM+(AP(I,K)*AP(K,KK)-ACP(I,K)*ACP(K,KK))
  111 DEN=(AP(I,I)*AP(I,I)+ACP(I,I)*ACP(I,I))
      IF(DEN)115,120,115
  115 CONR=AP(I,I)/DEN
      CONI=-ACP(I,I)/DEN
      AP(I,KK)=(CONR*AR(I,KK)-CONI*AC(I,KK))-(CONR*SUM-CONI*SAM)
      ACP(I,KK)=(CONI*AR(I,KK)+CONR*AC(I,KK))-(CONR*SAM+CONI*SUM)
      IF(NA-KK)120,101,114
  101 CONTINUE
  125 DO 116 I=1,NA
      SAM=0.0
      SUM=0.0
      IF(1-I)118,119,120
  118 KK=I-1
      DO 121 K=1,KK
      SUM=SUM+(AP(I,K)*CP(K)-ACP(I,K)*CCP(K))
  121 SAM=SAM+(AP(I,K)*CCP(K)+ACP(I,K)*CP(K))
  119 DEN=(AP(I,I)*AP(I,I)+ACP(I,I)*ACP(I,I))
      IF(DEN)122,120,122
  122 CONR=AP(I,I)/DEN
      CONI=-ACP(I,I)/DEN
      CP(I)=(CONR*C1(I)-CONI*CC(I))-(CONR*SUM-CONI*SAM)
  116 CCP(I)=(CONI*C1(I)+CONR*CC(I))-(CONI*SUM+CONR*SAM)
      XR(NA)=CP(NA)
```

```
      XI(NA)=CCP(NA)
      NB=NA-1
      DO 123 I2=1,NB
      I=NB-I2+1
      SAM=0.0
      SUM=0.0
      I3=I+1
      DO 124 K=I3,NA
      SUM=SUM+(AP(I,K)*XR(K)-ACP(I,K)*XI(K))
  124 SAM=SAM+(AP(I,K)*XI(K)+ACP(I,K)*XR(K))
      XR(I)=CP(I)-SUM
  123 XI(I)=CCP(I)-SAM
      RETURN
  130 PRINT 200
  200 FORMAT(19H0DETERMINANT = ZERO)
      RETURN
  120 PRINT 201
  201 FORMAT(6H0ERROR)
      RETURN
      END
```

APPENDIX C

NUMERICAL EXAMPLE

NUMERICAL EXAMPLE

In this Appendix we present a numerical example illustrating the programs described in Appendices A and B. Consider

$$P_{i1}\bar{\gamma}_0 + P_{i2}\bar{\varphi} + P_{i3}\bar{\xi}_1 + P_{i4}\bar{\xi}_2 = 0 \text{ , } i = 1,2,3,4,$$

where in the physical problem $\bar{\gamma}_0$, $\bar{\varphi}$, $\bar{\xi}_1$ and $\bar{\xi}_2$ are Laplace transforms of the following four degrees of freedom:

γ_0 = rigid body translation

φ = rigid body pitch

ξ_1 = first fuel sloshing

ξ_2 = second fuel sloshing

Let the polynomial elements of the matrix P be:

$$P_{11} = S + 7.1131409 \times 10^{-3}$$

$$P_{21} = 3.3673847 \times 10^{-5}S + 7.7574536 \times 10^{-4}$$

$$P_{31} = S$$

$$P_{41} = S$$

$$P_{12} = 1.3717434 \times 10^{-2}S^2 - 2.5805817 \times 10^{1}S - 4.3492353 \times 10^{1}$$

$$P_{22} = S^2 + 1.8100787\ S + 1.3966334$$

$$P_{32} = -\ 2.3408324 \times 10^{1}S^2 - 9.7847893$$

$$P_{42} = -\ 6.7790616\ S^2 - 9.7847893$$

$$P_{13} = 6.3417457 \times 10^{-2}S^2$$

$$P_{23} = -\ 3.6441732 \times 10^{-3}S^2 - 1.5232815 \times 10^{-3}$$

$$P_{33} = S^2 + 2.75 \times 10^{-1}S + 7.5625$$

$$P_{43} = 0$$

$$P_{14} = 9.9826665 \times 10^{-2}S^2$$

$$P_{24} = - 1.6612538 \times 10^{-3}S^2 - 2.3978274 \times 10^{-3}$$

$$P_{34} = 0$$

$$P_{44} = S^2 + 2.78 \times 10^{-1}S + 7.7283998$$

Program M0301 requires the matrix whose elements are coefficients of S^2 to be nonsingular. Without altering the spectrum of eigenvalues, it is possible to multiply the first column of matrix P by S. This is essentially equivalent to the system

$$SP_{i1}(\bar{\gamma}_o/S) + P_{i2}\bar{\varphi} + P_{i3}\bar{\xi}_1 + P_{i4}\bar{\xi}_2 = 0 \text{ , } i = 1,2,3,4 \text{ .}$$

Now, applying this conditioning factor to the polynomial elements of the matrix P given above, the input data for program M0301 are:

$$\underset{(\text{coefficients of } S^2)}{A} =$$

$$\begin{pmatrix} 1.0 & 1.3717434 \times 10^{-2} & 6.3417457 \times 10^{-2} & 9.9826665 \times 10^{-2} \\ 3.3673847 \times 10^{-5} & 1.0 & -3.6441732 \times 10^{-3} & -1.6612538 \times 10^{-3} \\ 1.0 & -2.3408324 \times 10^{1} & 1.0 & 0.0 \\ 1.0 & -6.7790616 & 0.0 & 1.0 \end{pmatrix}$$

$$\underset{\text{(coefficients of } S)}{B1} = \begin{pmatrix} 7.1131409 \times 10^{-3} & -2.5805817 \times 10^{1} & 0.0 & 0.0 \\ 7.7574536 \times 10^{-4} & 1.8100787 & 0.0 & 0.0 \\ 0.0 & 0.0 & 2.75 \times 10^{-1} & 0.0 \\ 0.0 & 0.0 & 0.0 & 2.78 \times 10^{-1} \end{pmatrix}$$

$$\underset{\text{(coefficients independent of } S)}{C} = \begin{pmatrix} 0.0 & -4.3492353 \times 10^{1} & 0.0 & 0.0 \\ 0.0 & 1.3966334 & -1.5232815 \times 10^{-3} & -2.3978274 \times 10^{-3} \\ 0.0 & -9.7847893 & 7.5625 & 0.0 \\ 0.0 & -9.7847893 & 0.0 & 7.7283998 \end{pmatrix}$$

In computing the corresponding eigenvectors in subprograms M0308 and M0309, the original unconditioned data are used. Matrices A1 (coefficients of S^2), B1 (coefficients of S) and C (coefficients independent of S) correspond, respectively, to matrices A, B1, and C given above with the exception that the first column in each of the above matrices must be replaced as follows.

$$\underset{A1}{\text{Column One}} = \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}, \quad \underset{B1}{\text{Column One}} = \begin{pmatrix} 1.0 \\ 3.3673847 \times 10^{-5} \\ 1.0 \\ 1.0 \end{pmatrix} \quad \text{and}$$

$$\begin{array}{c}\text{Column One} \\ C\end{array} = \begin{pmatrix} 7.1131409 \times 10^{-3} \\ 7.7574536 \times 10^{-4} \\ 0.0 \\ 0.0 \end{pmatrix}$$

Other input values necessary for this example are NN = 4 , LZ = 1 , NSO = 5 and five values of SS = -0.6, -0.4, -0.2, 0.0, 0.2.

The computed eigenvalues and corresponding eigenvectors based on these data are listed below. Execution time of this program for the given data on our 1620 IBM computer was approximately 15 min. Note that the execution time would be considerably reduced if one could use a faster computer with greater storage, e.g., an IBM 7090 or 360.

THE EIGENVALUES

-.0313530388752723	
-.9107409792013360	.7908257742484470
-.1510156405298613	2.7885845745208050
-.3588551179168403	3.0826959892069850

THE EIGENVALUE = -3.135303887527230E-02 0.000000000000000E-99

THE CORRESPONDING EIGENVECTOR

1	.1000000000000000E+01	.0000000000000000E-99
2	-.5678892331866050E-03	.0000000000000000E-99
3	.3412808668328876E-02	.0000000000000000E-99
4	.3340719024749410E-02	.0000000000000000E-99

THE EIGENVALUE = -9.107409792013360E-01 7.908257742484470E-01

THE CORRESPONDING EIGENVECTOR

1	-.2049164003771471E+01	-.2451688498958219E+02
2	.1000000000000000E+01	.0000000000000000E-99
3	.2802946774896149E+00	-.7195803054258117E+01
4	-.6659367603751580E+00	-.4074082319513876E+01

THE EIGENVALUE = -1.510156405298613E-01 2.788584574520805E+00

THE CORRESPONDING EIGENVECTOR

1	.2532377948525678E-01	-.6408295748643520E-01
2	-.2518910608894471E-02	.2571981414828410E-04
3	-.9607198912245160E+00	.4623725262937161E+00
4	.1000000000000000E+01	.0000000000000000E-99

THE EIGENVALUE = -3.588551179168403E-01 3.082695989206985E+00

THE CORRESPONDING EIGENVECTOR

1	.2799901067746250E+00	-.5343778898294774E+00
2	.4876980626030941E-02	.4230670695054194E-02
3	.1357835994917070E+01	.1786710202015887E+00
4	.1000000000000000E+01	.0000000000000000E-99